AI Content Detection Using RNNs

Moin Arz Mattar
Dartmouth College - Class of 2028 (freshman year)
moin.mattar1@gmail.com
Professor Soroush Vosoughi

March 8, 2025

Abstract

Distinguishing between human-written and AI-generated text is becoming increasingly important in the face of rapidly evolving Large Language Models (LLMs). This paper presents a novel approach that leverages two closely related models, Falcon-7B and Falcon-7B-Instruct, to compute token-wise log perplexity and cross-perplexity. These values are then used to train a classifier that can detect AI-generated text. Our results demonstrate robust performance on various datasets, suggesting this method is a promising direction for next-generation AI detection.

1 Introduction

With the rapid advancement of LLMs, distinguishing between human-written and AI-generated text has become a crucial task. Many existing approaches rely on statistical analysis or supervised learning models, but we propose a novel approach leveraging RNNs to process token-wise perplexity and cross-entropy values computed from two LLMs. The goal of this work is to create an AI detection method that is both accurate and computationally efficient, ensuring that even the most sophisticated AI-generated text can be reliably identified.

2 Methodology

2.1 Data Preparation

To ensure a robust AI detector, we compiled a comprehensive dataset consisting of both AI-generated and human-written text. The datasets used include:

- `1st_place_in_llm_detection_competition`
- `ChatGPT-Detector-Bias`
- `CHEAT`

- `CheckGPT-main`
- `fineweb-edu`
- `ghostbuster-data`
- `gpt2-output-data`
- `HC3`, `M4-main`, `open-text-books`
- `OriginalityAI-benchmark`, `OUTFOX`, `SimPajama-llm-comp`, `Tweepfake`

These datasets span multiple topics, styles, and lengths, allowing the model to learn to detect AI-generated content across different domains and writing styles.

2.2 Contrastive Perplexity Analysis

We employ two LLMs, denoted as M_1 and M_2. For each token in a text sequence, we calculate:

$$\log \mathrm{PPL}_{M_1}(s_i) = \log\big(P_{M_1}(x_i)\big),$$

where $P_{M_1}(x_i)$ is the probability assigned by model M_1 to the ith token x_i. We further define:

$$\log X\text{-PPL}_{M_1,M_2}(s_i) = P_{M_1}(x_i) \cdot \log\big(P_{M_2}(x_i)\big).$$

These token-wise metrics provide insight into how similarly (or differently) the two models predict each token. This *contrastive* setup can highlight subtle distinctions between AI-generated and human text.

2.3 Model Architecture

We use these token-level features as inputs to a classifier. In our experiments, we primarily use a recurrent neural network (RNN) but note that other architectures could be used. The token-wise nature of our feature extraction is maintained throughout, allowing the sequential model to capture patterns that might be missed by simply summing values across an entire sequence.

3 Results

We trained our detector on the aforementioned datasets and evaluated its performance. Figure 1 shows the training and validation accuracy, while Figure 2 shows the corresponding loss curves. We also tracked the learning rate schedule (Figure 3), and present the ROC curve (Figure 4) to illustrate the model's ability to distinguish positive (AI-generated) from negative (human-written) examples.

Overall, the model achieved:

- Training and validation accuracy: 85–90%

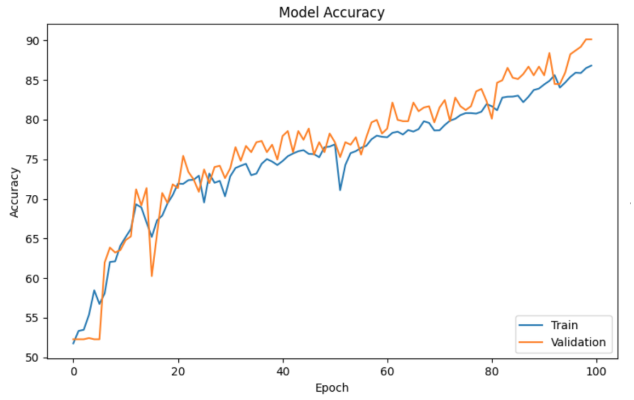

Figure 1: Training and validation accuracy.

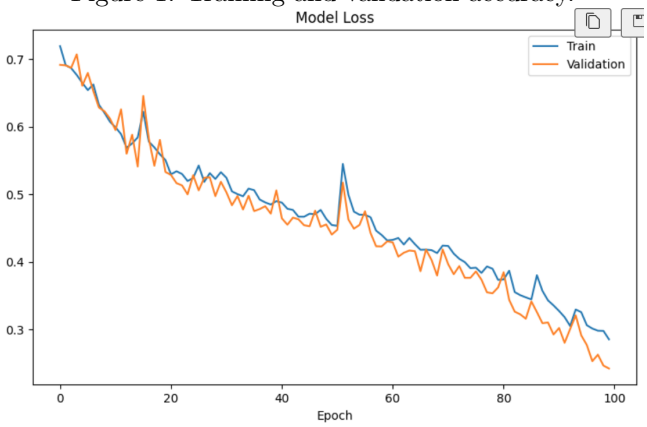

Figure 2: Training and validation loss.

- Test accuracy: 75%
- Area Under Curve (AUC) of approximately 0.82

These results suggest that contrastive perplexity signals, when modeled by an RNN, provide a strong signal for AI text detection.

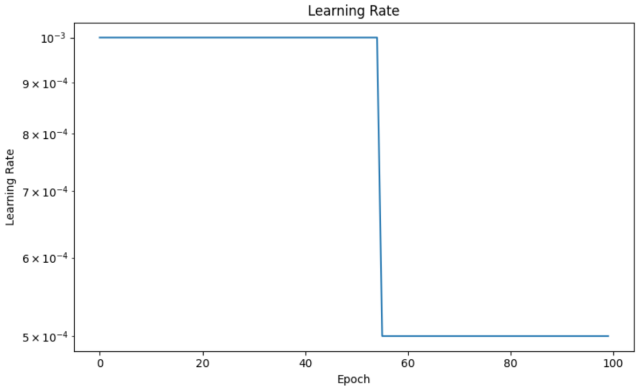

Figure 3: Learning rate schedule over epochs.

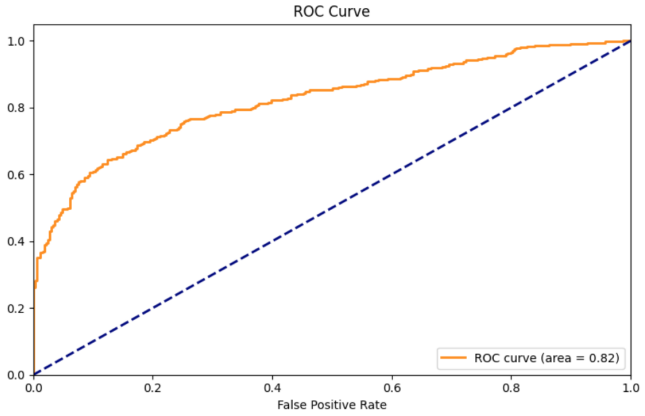

Figure 4: ROC curve on the test set, illustrating the trade-off between true positive rate and false positive rate.

4 RNN Fundamentals

Recurrent Neural Networks (RNNs) are particularly effective in tasks involving sequential data. Unlike feedforward networks, RNNs maintain a hidden state $\mathbf{h}_t$ that is updated at each time step t:

$$\mathbf{h}_t = f(\mathbf{W}_h \mathbf{h}_{t-1} + \mathbf{W}_x \mathbf{x}_t),$$

where $\mathbf{x}_t$ is the current input (our token-wise features), and $\mathbf{h}_{t-1}$ is the previous hidden state. Figure 5 depicts a simplified RNN unrolled over multiple time steps.

By processing log perplexity and cross-perplexity features over the token sequence, the RNN can accumulate evidence and detect patterns indicative of AI or human text.

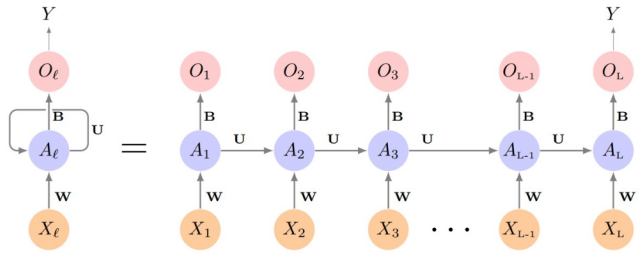

Figure 5: Unrolled representation of a basic RNN architecture.

5 Logits in LLMs

In an LLM, *logits* are the unnormalized outputs for each token in the vocabulary. Formally, if the model output vector is $\mathbf{z}$, the probability of token v_j is:

$$P(v_j) = \frac{\exp(\mathbf{z}_j)}{\sum_k \exp(\mathbf{z}_k)}.$$

Examining logits can provide insight into the model's confidence across different tokens. In our experiments, we often look at these logits to understand how two models might diverge in their token predictions.

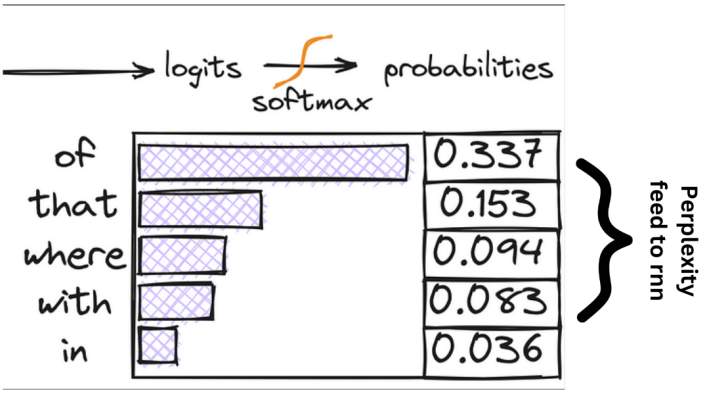

Figure 6: Illustration of model logits before applying the softmax function.

6 Existing Solutions

Existing AI-text detection approaches can be broadly categorized into:

- **Statistical Methods**: Simple metrics like perplexity or stylometric features.

- **Classifier-Based Methods**: Training a model on known AI-generated and human text.

The *Binoculars* method, for example, sums log perplexity and cross-entropy values over entire sequences to produce a single score. In contrast, our approach retains the token-wise perplexity and cross-perplexity signals as a sequence. By processing these signals with an RNN, we can capture nuanced changes in how the text is generated, leading to more robust detection performance.

7 Conclusion

Our contrastive approach, using perplexities from two LLMs and an RNN for sequence modeling, provides a promising direction for AI-generated text detection. Future work includes:

- Extending our method to newer and more diverse LLMs.
- Investigating robustness against adversarially generated text.
- Exploring Transformer-based encoders for token-wise features.

Project Links

For further details, code, and additional experiments, please visit:

- **Project Link:** https://v0-cs74-orcin.vercel.app/
- **GitHub Repository:** https://github.com/mo-root/CS74Project

References

1. Hans, A., Schwarzschild, A., Cherepanova, V., Kazemi, H., Saha, A., Goldblum, M., Geiping, J., Goldstein, T. (2024). Spotting LLMs With Binoculars: Zero-Shot Detection of Machine-Generated Text. arXiv preprint arXiv:2401.12070.